

NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCCCCCCCCCCC	PPPPPPPPPPPP	
NNN		NNN	CCC	PPP	PPP
NNN		NNN	CCC	PPP	PPP
NNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNNNNN		NNN	CCC	PPP	PPP
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN	NNN	NNN	CCC	PPPPPPPPPPPP	
NNN		NNNNNN	CCC	PPP	
NNN		NNNNNN	CCC	PPP	
NNN		NNNNNN	CCC	PPP	
NNN		NNN	CCC	PPP	
NNN		NNN	CCC	PPP	
NNN		NNN	CCC	PPP	
NNN		NNN	CCC	PPP	
NNN		NNN	CCCCCCCCCCCC	PPP	
NNN		NNN	CCCCCCCCCCCC	PPP	
NNN		NNN	CCCCCCCCCCCC	PPP	

.....

```

NN      NN      CCCCCCCC  PPPPPPPP  SSSSSSSS  TTTTTTTTTT  AAAAAA  ZZZZZZZZZZ  EEEEEEEEEE  RRRRRRRR
NN      NN      CCCCCCCC  PPPPPPPP  SSSSSSSS  TTTTTTTTTT  AAAAAA  ZZZZZZZZZZ  EEEEEEEEEE  RRRRRRRR
NN      NN      CC        PP        PP  SS        TT        AA        AA        EE        RR
NN      NN      CC        PP        PP  SS        TT        AA        AA        EE        RR
NNNN    NN      CC        PP        PP  SS        TT        AA        AA        EE        RR
NNNN    NN      CC        PP        PP  SS        TT        AA        AA        EE        RR
NN      NN      CC        PPPPPPPP  SSSSSS    TT        AA        AA        EEEEEEEE  RRRRRRRR
NN      NN      CC        PPPPPPPP  SSSSSS    TT        AA        AA        EEEEEEEE  RRRRRRRR
NN      NNNN    CC        PP        SS        TT        AAAAAAAAAA  ZZ        EE        RR
NN      NNNN    CC        PP        SS        TT        AAAAAAAAAA  ZZ        EE        RR
NN      NN      CC        PP        SS        TT        AA        AA        ZZ        RR
NN      NN      CC        PP        SS        TT        AA        AA        ZZ        RR
NN      NN      CCCCCCCC  PP        SSSSSSSS  TT        AA        AA        ZZZZZZZZZZ  EEEEEEEEEE  RR
NN      NN      CCCCCCCC  PP        SSSSSSSS  TT        AA        AA        ZZZZZZZZZZ  EEEEEEEEEE  RR

```

```

LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLLLL IIIIII  SSSSSSSS
LLLLLLLLLLLL IIIIII  SSSSSSSS

```

```
0001 0 %TITLE 'Zero Parse States and Data'
0002 0 MODULE NCPSTAZER (IDENT = 'V04-000', LIST(NOOBJECT)) =
0003 1 BEGIN
0004 1
0005 1
0006 1 *****
0007 1 *
0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0010 1 * ALL RIGHTS RESERVED.
0011 1 *
0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0017 1 * TRANSFERRED.
0018 1 *
0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0021 1 * CORPORATION.
0022 1 *
0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0025 1 *
0026 1 *
0027 1 *****
0028 1
0029 1
0030 1 ++
0031 1 FACILITY:      Network Control Program (NCP)
0032 1
0033 1 ABSTRACT:
0034 1
0035 1     States and data for the parsing of NCP zero commands
0036 1
0037 1 ENVIRONMENT:   VAX/VMS Operating System
0038 1
0039 1 AUTHOR:        Darrell Duffy      , CREATION DATE: 25-September-79
0040 1
0041 1 MODIFIED BY:
0042 1
0043 1     V03-003  RPG0003      Bob Grosso      09-Nov-1982
0044 1             Accept just ZERO X25-PROT without DTE or K DTE.
0045 1
0046 1     V03-002  RPG0002      Bob Grosso      07-Sep-1982
0047 1             Break up into several PCLs and supply an SDB for
0048 1             each entity type to facilitate correct error reporting
0049 1             of entity type. Also correct Zero Module X25-Protocol
0050 1             and accept Module X25-Server and X29-Server.
0051 1
0052 1     V03-001  RPG0001      Bob Grosso      03-Aug-1982
0053 1             Add ZERO MODULE X25-PROTOCOL
0054 1
0055 1     V001     TMH0001      Tim Halvorsen   22-Jun-1981
0056 1             Change BUILD_SDB reference to use full entity type field.
0057 1             Change to zero circuit as well as line and node.
```


NCPSTAZER
V04-000

Zero Parse States and Data

; 58

0058 1 !--

K 10
16-Sep-1984 01:47:08
14-Sep-1984 12:48:33

VAX-11 Bliss-32 V4.0-742
[NCP.SRC]NCPSTAZER.B32;1

Page 2
(1)

NCI
VOI

.....

NCPSTAZER
V04-000

Zero Parse States and Data Definitions

L 10
16-Sep-1984 01:47:08
14-Sep-1984 12:48:33

VAX-11 Bliss-32 V4.0-742
[NCP.SRC]NCPSTAZER.B32;1

Page 3
(2)

```

60      0059 1 %SBTTL 'Definitions'
61      0060 1
62      0061 1
63      0062 1
64      0063 1
65      0064 1
66      0065 1
67      0066 1
68      0067 1
69      0068 1
70      0069 1
71      0070 1
72      0071 1
73      0072 1
74      0073 1

```

! External symbols for action routines

```
.. 76      0074 1 %SBTTL 'Parameter blocks'
.. 77      0075 1
.. 78      0076 1
.. 79      0077 1 BIND DATA:
.. 80      0078 1
.. 81      0079 1
.. 82      0080 1
.. 83      0081 1 Parameter Blocks
.. 84      0082 1
.. 85      0083 1
.. 86      0084 1
.. 87      0085 1 General use for entities which do not take parameters
.. 88      0086 1
.. 89      0087 1
.. 90      P 0088 1 BUILD_PCL
.. 91      P 0089 1
.. 92      P 0090 1 (ZER,
.. 93      P 0091 1
.. 94      P 0092 1 , END, , ,
.. 95      P 0093 1
.. 96      0094 1 )
.. 97      0095 1
.. 98      P 0096 1 BUILD_PBK
.. 99      P 0097 1
100      P 0098 1 (ZER,
101      P 0099 1
102      P 0100 1 EXE, LITL, 0, VRB_ENT,
103      P 0101 1 CIR, TKN, , VRB_ENT,
104      P 0102 1 LIN, TKN, , VRB_ENT,
105      P 0103 1 MOD, TKN, , VRB_ENT,
106      P 0104 1 NOD, NADR, , VRB_ENT,
107      P 0105 1 KNO, LITB, NMASC_ENT_KNO, VRB_ENT,
108      P 0106 1
109      0107 1 )
110      0108 1
111      P 0109 1 BUILD_SDB
112      P 0110 1
113      0111 1 (ZCI, NMASC_ENT_CIR, VRB_ENT, ZER)
114      0112 1
115      0113 1
116      P 0114 1 BUILD_SDB
117      P 0115 1
118      0116 1 (ZLI, NMASC_ENT_LIN, VRB_ENT, ZER)
119      0117 1
120      0118 1
121      P 0119 1 BUILD_SDB
122      P 0120 1
123      0121 1 (ZMO, NMASC_ENT_MOD, VRB_ENT, ZER)
124      0122 1
125      0123 1
126      P 0124 1 BUILD_SDB
127      P 0125 1
128      0126 1 (ZNO, NMASC_ENT_NOD, VRB_ENT, ZER)
.. 129      0127 1
```



```
.. 131      0128 1  !
.. 132      0129 1  !
.. 133      0130 1  !
.. 134      0131 1  !
.. 135      P 0132 1  BUILD_PCL
.. 136      P 0133 1  (ZPR,
.. 137      P 0134 1  DTE, TKN, PCXP_DTE, ,
.. 138      P 0135 1  { END, . . ,
.. 139      P 0136 1  }
.. 140      P 0137 1  BUILD_PBK
.. 141      P 0138 1  (ZPR,
.. 142      P 0139 1  DTE, TKN,
.. 143      P 0140 1  KDT, LITB, 'NMA$C_ENT_KNO, ZPR_DTE,      ! Known DTEs
.. 144      P 0141 1  )
.. 145      P 0142 1  BIND PDB$G_ZPR_ENT = UPLIT BYTE (0, %ASCIC 'X25-PROTOCOL');
.. 146      P 0143 1  BUILD_SDB
.. 147      P 0144 1  (ZPR, NMA$C_ENT_MOD, ZPR_ENT, ZPR)
.. 148      P 0145 1
.. 149      P 0146 1
.. 150      P 0147 1
.. 151      P 0148 1
.. 152      P 0149 1
.. 153      P 0150 1
.. 154      P 0151 1
.. 155      P 0152 1
.. 156      P 0153 1
.. 157      P 0154 1
.. 158      P 0155 1
.. 159      P 0156 1
.. 160      P 0157 1  !
.. 161      P 0158 1  !
.. 162      P 0159 1  !
.. 163      P 0160 1  BIND PDB$G_ZSE_ENT = UPLIT BYTE (0, %ASCIC 'X25-SERVER');
.. 164      P 0161 1  BUILD_SDB
.. 165      P 0162 1  (ZSE, NMA$C_ENT_MOD, ZSE_ENT, ZER)
.. 166      P 0163 1
.. 167      P 0164 1
.. 168      P 0165 1
.. 169      P 0166 1
.. 170      P 0167 1  !
.. 171      P 0168 1  !
.. 172      P 0169 1  !
.. 173      P 0170 1  !
.. 174      P 0171 1  BIND PDB$G_Z9S_ENT = UPLIT BYTE (0, %ASCIC 'X29-SERVER');
.. 175      P 0172 1  BUILD_SDB
.. 176      P 0173 1  (Z9S, NMA$C_ENT_MOD, Z9S_ENT, ZER)
.. 177      P 0174 1
.. 178      P 0175 1
```

```
: 180      0176 1 %SBTTL 'Prompt strings'
: 181      0177 1
: 182      0178 1
: 183      0179 1
: 184      0180 1
: 185      0181 1
: 186      0182 1
: 187      0183 1
: 188      P 0184 1
: 189      P 0185 1
: 190      P 0186 1
: 191      L 0187 1
: 192      L 0188 1
: 193      P 0189 1
: 194      P 0190 1
: 195      P 0191 1
: 196      P 0192 1
: 197      P 0193 1
: 198      P 0194 1
: 199      P 0195 1
: 200      0196 2
: 201      0197 1
: 202      0198 1
: 203      P 0199 1
: 204      P 0200 1
: 205      P 0201 1
: 206      P 0202 1
: 207      P 0203 1
: 208      0204 2
: 209      0205 1
: 210      0206 1

      1 %SBTTL 'Prompt strings'
      |
      | Build prompt strings
      |
      | BIND
      |
      | PROMPT_STRINGS
      | (ZER,
      |
      | ENT, %STRING(
      |   (EXECUTOR, CIRCUIT circuit-name, LINE line-id, ', CRLF,
      |   MODULE module-name, NODE node-id, KNOWN): '),
      |   CIR, 'Circuit to zero (circuit-name): ',
      |   LIN, 'Line to zero (dev-c-u.t): ',
      |   MOD, 'Module to zero (X25-PROTOCOL, X25-SERVER, X29-SERVER): ',
      |   NOD, 'Node to zero (node-name, address): ',
      |   KWN, ' (CIRCUITS, LINES, MODULES, NODES): ',
      | )
      | ,
      |
      | PROMPT_STRINGS
      | (ZPR,
      |
      | DAT, '(DTE dte-name, KNOWN DTES): ',
      | )
      |
      |
```



```
212 0207 1 %SBTTL 'State Table Entry'
213 0208 1
214 0209 1 $INIT_STATE (NCP$G_STTBL_ZER, NCP$G_KYTBL_ZER);
215 0210 1
216 0211 1
217 0212 1 Zero Command
218 0213 1
219 0214 1
220 0215 1
221 0216 1 Dispatch to entity types and prompt if end of string
222 0217 1
223 0218 1
224 P 0219 1 COMMAND PROMPT
225 P 0220 1 (ZER, ENT, NCP$ _INVKEY,
226 P 0221 1
227 P 0222 1 ('EXECUTOR', ST_ZNO_DOIT, ACT$SAVPRM, , PBK$G_ZER_EXE),
228 P 0223 1 ('KNOWN', ST_ZER_KWN, ACT$SAVPRM, , , PBK$G_ZER_KNO),
229 P 0224 1 ('CIRCUIT', ST_ZER_CIR),
230 P 0225 1 ('LINE', ST_ZER_LIN),
231 P 0226 1 ('MODULE', ST_ZER_MOD),
232 P 0227 1 ('NODE', ST_ZER_NOD),
233 P 0228 1 ('X25', ST_ZER_X25),
234 P 0229 1 ('X29', ST_ZER_X29)
235 P 0230 1
236 0231 1 )
237 0232 1
238 0233 1
239 0234 1 Obtain the circuit to zero
240 0235 1
241 0236 1
242 P 0237 1 COMMAND PROMPT
243 P 0238 1 (ZER, CIR, NCP$ _INVVAL,
244 P 0239 1
245 P 0240 1 ( (SE_CIRC_ID), ST_ZCI_DOIT, ACT$SAVPRM, NMASC_ENT_CIR,
246 P 0241 1 NCP$GL_OPTION, PBK$G_ZER_CIR)
247 P 0242 1
248 0243 1 )
249 0244 1
250 0245 1 Perform the function after ignoring noise words
251 0246 1
252 0247 1
253 P 0248 1 $STATE (ST_ZCI_DOIT,
254 P 0249 1 ('COUNTERS'),
255 P 0250 1 (TPAS_LAMBDA)
256 0251 1 );
257 0252 1
258 P 0253 1 $STATE (, ! Allow for no parameters
259 P 0254 1 (TPAS_LAMBDA, , ACT$SAVPRM, , , PBK$G_VRB_ALL)
260 0255 1 );
261 0256 1
262 P 0257 1 $STATE (,
263 P 0258 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_UTILITY, , , SDB$G_ZCI)
264 0259 1 );
```

```
.. 266      0260 1 |  
.. 267      0261 1 | Obtain the line to zero  
.. 268      0262 1 |  
.. 269      0263 1 |  
.. 270      P 0264 1 |  
.. 271      P 0265 1 | COMMAND PROMPT  
.. 272      P 0266 1 | (ZER, LIN, NCP$_INVVAL,  
.. 273      P 0267 1 | ( (SE_LINE_ID), ST_ZLI_DOIT, ACT$SAVPRM, NMASC_ENT_LIN,  
.. 274      P 0268 1 | NCP$GL_OPTION, PBK$G_ZER_LIN)  
.. 275      P 0269 1 |  
.. 276      0270 1 | )  
.. 277      0271 1 |  
.. 278      0272 1 | Perform the function after ignoring noise words  
.. 279      0273 1 |  
.. 280      0274 1 |  
.. 281      P 0275 1 | $STATE (ST_ZLI_DOIT,  
.. 282      P 0276 1 | ('COUNTERS'),  
.. 283      P 0277 1 | (TPAS_LAMBDA)  
.. 284      0278 1 | );  
.. 285      0279 1 |  
.. 286      P 0280 1 | $STATE (,  
.. 287      P 0281 1 | (TPAS_LAMBDA, , ACT$SAVPRM, , , PBK$G_VRB_ALL)  
.. 288      0282 1 | );  
.. 289      0283 1 |  
.. 290      P 0284 1 | $STATE (,  
.. 291      P 0285 1 | (TPAS_EOS, TPAS_EXIT, ACT$VRB_UTILITY, , , SDB$G_ZLI)  
.. 292      0286 1 | );
```



```
294 0287 1 |
295 0288 1 | Obtain the module to zero
296 0289 1 |
297 0290 1 |
298 P 0291 1 $STATE (ST_ZER_MOD,
299 P 0292 1 ('X25', ST_ZER_X25),
300 P 0293 1 ('X29', ST_ZER_X29),
301 P 0294 1 (TPAS_EOS, ST_ZER_PMT_MOD),
302 0295 1 );
303 0296 1
304 P 0297 1 $STATE (ST_ZER_PMT_MOD,
305 0298 1 (TPAS_LAMBDA, , ACT$PRMPT, , PMT$G_ZER_MOD));
306 0299 1
307 P 0300 1 $STATE (
308 P 0301 1 ('X25', ST_ZER_X25),
309 P 0302 1 ('X29', ST_ZER_X29),
310 P 0303 1 (TPAS_EOS, ST_ZER_PMT_MOD)
311 0304 1 );
312 0305 1
313 P 0306 1 $STATE (ST_ZER_X25,
314 0307 1 ('-'));
315 P 0308 1 $STATE (
316 P 0309 1 ('PROTOCOL', ST_ZPR_GO, , NMASC_ENT_MOD, NCP$GL_OPTION),
317 P 0310 1 ('SERVER', ST_ZSE_DOIT, , NMASC_ENT_MOD, NCP$GL_OPTION)
318 0311 1 );
```



```

320 0312 1
321 0313 1
322 0314 1
323 0315 1
324 0316 1
325 P 0317 1 $STATE (ST_ZPR_GO,
326 P 0318 1 (TPAS_EOS, ST_ZPR_PMT_DAT),
327 P 0319 1 (TPAS_LAMBDA, ST_ZPR_DAT)
328 0320 1 );
329 0321 1
330 P 0322 1 $STATE (ST_ZPR_PMT_DAT,
331 0323 1 (TPAS_LAMBDA, ACT$PRMPT, PMT$G_ZPR_DAT));
332 0324 1
333 P 0325 1 $STATE (ST_ZPR_DAT,
334 P 0326 1 ('DTE', ST_ZPR_DAT_DTE),
335 P 0327 1 ('KNOWN', ST_ZPR_DAT_KWN),
336 P 0328 1 (TPAS_LAMBDA, ST_ZPR_DOIT)
337 0329 1 );
338 0330 1
339 P 0331 1 $STATE (ST_ZPR_DAT_DTE,
340 0332 1 ((SE_DTE_NUMBER), ST_ZPR_DOIT, ACT$SAVPRM, PBK$G_ZPR_DTE));
341 0333 1
342 P 0334 1 $STATE (ST_ZPR_DAT_KWN,
343 0335 1 ('DTE$', ST_ZPR_DOIT, ACT$SAVPRM, PBK$G_ZPR_KDT));
344 0336 1
345 0337 1
346 0338 1
347 0339 1
348 0340 1
349 P 0341 1 $STATE (ST_ZPR_DOIT,
350 P 0342 1 ('COUNTERS'),
351 P 0343 1 (TPAS_LAMBDA)
352 0344 1 );
353 0345 1
354 P 0346 1 $STATE (, ! Allow for no parameters
355 P 0347 1 (TPAS_LAMBDA, , ACT$SAVPRM, , PBK$G_VRB_ALL)
356 0348 1 );
357 0349 1
358 P 0350 1 $STATE (,
359 0351 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_UTILITY, , SDB$G_ZPR));
```

```

361 0352 1
362 0353 1
363 0354 1 X25-SERVER
364 0355 1
365 0356 1
366 0357 1
367 0358 1 Perform the function after ignoring noise words
368 0359 1
369 0360 1
370 P 0361 1 $STATE (ST_ZSE_DOIT,
371 P 0362 1 ('COUNTERS'),
372 P 0363 1 (TPAS_LAMBDA)
373 0364 1 );
374 0365 1
375 P 0366 1 $STATE (
376 P 0367 1 (TPAS_LAMBDA, , ACT$SAVPRM, , , PBK$G_VRB_ALL)
377 0368 1 );
378 0369 1
379 P 0370 1 $STATE (
380 P 0371 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_UTILITY, , , SDB$G_ZSE)
381 0372 1 );
382 0373 1
383 0374 1
384 0375 1
385 0376 1 X29-SERVER
386 0377 1
387 0378 1
388 P 0379 1 $STATE (ST_ZER_X29,
389 0380 1 ('-T'));
390 P 0381 1 $STATE (
391 P 0382 1 ('SERVER', ST_Z9S_DOIT,, NMASC_ENT_MOD, NCP$GL_OPTION)
392 0383 1 );
393 0384 1
394 0385 1
395 0386 1 Perform the function after ignoring noise words
396 0387 1
397 0388 1
398 P 0389 1 $STATE (ST_Z9S_DOIT,
399 P 0390 1 ('COUNTERS'),
400 P 0391 1 (TPAS_LAMBDA)
401 0392 1 );
402 0393 1
403 P 0394 1 $STATE (
404 P 0395 1 (TPAS_LAMBDA, , ACT$SAVPRM, , , PBK$G_VRB_ALL)
405 0396 1 );
406 0397 1
407 P 0398 1 $STATE (
408 P 0399 1 (TPAS_EOS, TPAS_EXIT, ACT$VRB_UTILITY, , , SDB$G_Z9S)
409 0400 1 );
```

```

: 411      0401 1 |
: 412      0402 1 | Obtain the node to zero
: 413      0403 1 |
: 414      0404 1 |
: 415      P 0405 1 | COMMAND PROMPT
: 416      P 0406 1 | (ZER, NOD, NCP$_INVVAL,
: 417      P 0407 1 |
: 418      P 0408 1 | ( (SE_NODE_ID), ST_ZNO_DOIT, ACT$SAVPRM, NMASC_ENT NOD,
: 419      P 0409 1 | NCP$GL_OPTION, PBK$G_ZER_NOD)
: 420      P 0410 1 |
: 421      0411 1 | )
: 422      0412 1 |
: 423      0413 1 | Perform the function after ignoring noise words
: 424      0414 1 |
: 425      0415 1 |
: 426      P 0416 1 | $STATE (ST_ZNO_DOIT,
: 427      P 0417 1 | ('COUNTERS'),
: 428      P 0418 1 | (TPAS_LAMBDA)
: 429      0419 1 | );
: 430      0420 1 |
: 431      P 0421 1 | $STATE (, ! Allow for no parameters
: 432      P 0422 1 | (TPAS_LAMBDA, , ACT$SAVPRM, , , PBK$G_VRB_ALL)
: 433      0423 1 | );
: 434      0424 1 |
: 435      P 0425 1 | $STATE (,
: 436      P 0426 1 | (TPAS_EOS, TPAS_EXIT, ACT$VRB_UTILITY, , , SDB$G_ZNO)
: 437      0427 1 | );
```



```
.. 439      0428 1
.. 440      0429 1
.. 441      0430 1
.. 442      0431 1
.. 443      0432 1
.. 444      P 0433 1
.. 445      P 0434 1
.. 446      P 0435 1
.. 447      P 0436 1
.. 448      P 0437 1
.. 449      P 0438 1
.. 450      P 0439 1
.. 451      P 0440 1
.. 452      0441 1
.. 453      0442 1
.. 454      0443 1
.. 455      0444 1
.. 456      0445 1
.. 457      0446 1
.. 458      P 0447 1
.. 459      P 0448 1
.. 460      P 0449 1
.. 461      0450 1
.. 462      0451 1
.. 463      P 0452 1
.. 464      P 0453 1
.. 465      0454 1
.. 466      0455 1
.. 467      P 0456 1
.. 468      P 0457 1
.. 469      0458 1

Find out which is known, lines or nodes

COMMAND PROMPT
(ZER, KQN, NCP$_INVKEY,

('CIRCUITS', ST_ZCI_DOIT, , NMASC_ENT CIR, NCP$GL_OPTION),
('LINES', ST_ZLI_DOIT, , NMASC_ENT LIN, NCP$GL_OPTION),
('MODULES', NMASC_ENT MOD, NCP$GL_OPTION),
('NODES', ST_ZNO_DOIT, , NMASC_ENT_NOD, NCP$GL_OPTION)
)

Perform the function after ignoring noise words

$STATE (
('COUNTERS'),
(TPAS_LAMBDA)
);

$STATE (
, ! Allow for no parameters
(TPAS_LAMBDA, , ACT$SAVPRM, , , PBK$G_VRB_ALL)
);

$STATE (
,
(TPAS_EOS, TPAS_EXIT, ACT$VRB_UTILITY, , , SDB$G_ZMO)
);
```

```

: 471 0459 1 %SBTTL 'Define Subexpressions from Library'
: 472 0460 1
: 473 0461 1 !
: 474 0462 1 ! Define subexpressions from library
: 475 0463 1 !
: 476 0464 1
: 477 0465 1 SEM_DTE_NUMBER ! DTE number
: 478 0466 1 SEM_LINE_ID ! Line id strings
: 479 0467 1 SEM_CIRC_ID ! Circuit name strings
: 480 0468 1 SEM_NODE_ID ! Node id strings

```

NCPSTAZER
V04-000

Zero Parse States and Data
Object Listing of Parse Table

K 11
16-Sep-1984 01:47:08
14-Sep-1984 12:48:33

VAX-11 Bliss-32 V4.0-742
[NCP.SRC]NCPSTAZER.B32;1

Page 15
(14)

:	482	0469	1	%SBTTL	'Object Listing of Parse Table'	
:	483	0470	1			
:	484	0471	1	END		!End of module
:	485	0472	0	ELUDOM		

0271

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY